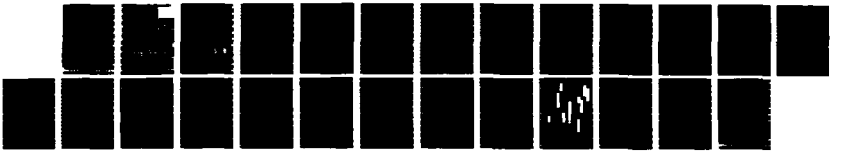
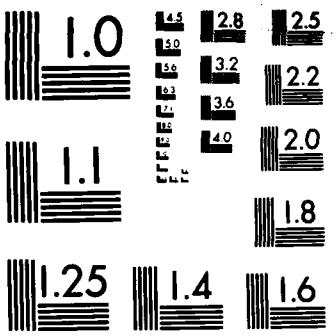
AD-A174 695 INTERACTIVE OPTIMIZATION METHODOLOGY FOR FLEET SCHEDULING(U) GEORGIA INST OF TECH ATLANTA PRODUCTION AND DISTRIBUTION RESEARCH CENTER W G NULTY ET AL 1/1

UNCLASSIFIED SEP 86 PDRC-86-11 N00014-86-K-0173 F/G 13/10 NL





INTERACTIVE OPTIMIZATION METHODOLOGY FOR FLEET SCHEDULING

by

William G. Nulty
H. Donald Ratliff

PDRC 86-11

13

PDRC Report Series 86-11
September 1986

INTERACTIVE OPTIMIZATION METHODOLOGY FOR FLEET SCHEDULING

by

William G. Nulty
H. Donald Ratliff

PDRC 86-11

School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

This work was supported in part by the Office of Naval Research under Contract No N00014-86-K-0173. Reproduction in whole or in part is permitted for any purpose of the U. S. Government.

Abstract

This paper addresses the problem of scheduling the United States Navy's Atlantic Fleet to overseas strategic requirements. The requirements are unique in a scheduling context in that the start and stop times to process them are fixed in advance. An integer programming formulation for the problem is developed. However, the integer program is too large to optimally solve. This fact and the subjective nature of additional secondary objectives and constraints suggest an interactive optimization approach. A system which solves a relaxation of the integer program within an interactive environment is discussed.

1.0 PROBLEM DEFINITION

The scheduling of ships to overseas strategic requirements for the United States Navy's Atlantic Fleet is a challenging optimization problem. The problem is difficult from a mathematical perspective because of a complex constraint structure. The problem is difficult from a philosophical perspective because some of the objectives and constraints are not precisely defined, or are difficult-to-quantify.

A deployment requirement is a need for a ship with certain characteristics to visit a location over some time period (e.g., a destroyer with helicopter capability is required in the Mediterranean from July 1 to December 15). The start and stop times for each requirement are fixed in advance. The primary objective of the problem is to satisfy all the requirements with the available ships.

The fundamental constraints on the problem are:

1. A requirement can be assigned to only one ship.
2. At any time a ship can process only one requirement.
3. A requirement can be assigned to a ship if the ship is available and if the ship has the required characteristics.
4. A minimum time may be required between the completion of one requirement by a ship and the beginning of the next requirement by the same ship (e.g., the ship must be in port six months between deployments). In addition, this constraint may be requirement dependent.
5. Some requirements may be prohibited from following other requirements on the same ship (e.g, a ship might not be allowed to have two particularly undersirable deployments in succession).

6. Ships are out of service for specified periods due to overhaul.

A very special case of this problem is the well known "Tanker Scheduling Problem" solved by Dantzig and Fulkerson [1]. This is the problem which would result if all ships were identical and all ships were available during the entire planning horizon.

The problem also contains various secondary objectives which are not easily quantifiable. Examples of these secondary objectives include the following:

1. Balance the ship assignments so the fractions of time ships are away from their home ports are roughly equal.
2. Balance the ship assignments so ships are not away from home ports for two consecutive Christmas seasons.
3. Balance the ship assignments so that ships get different types of desirable training.

Additional factors which impact the problem include:

1. Some requirements may already be assigned to ships.
2. Atlantic Fleet deployment continues indefinitely, but the naval planners have knowledge only over some finite horizon. Uncertainty exists with regard to requirements and resources in the future.

These properties and the secondary objectives are best addressed by a experienced naval planner who utilizes an inherent set of strategic values to assess the quality of a schedule. This fact and the complexity of the mathematical optimization formulation suggest an interactive optimization approach.

1.1 Interactive Optimization Fundamentals

Interactive optimization exploits the user's ability to address the difficult-to-quantify issues, while utilizing the computer to perform the necessary complex numerical calculations. There are three fundamental components of an interactive optimization system: (1) the underlying mathematical models which aid in the solution proves; (2) the methodology and level of user interaction: (3) the interactive interface between user and computer.

1.1.1 Underlying Mathematical Models

The algorithms for the underlying mathematical model address the quantifiable issues and provide a reasonable solution to the problem. Algorithms are necessary because the user typically cannot generate an acceptable solution because of the complex requirements and the huge number of possible solutions which might require evaluation. Low-level tactical decisions are made by the algorithms.

1.1.2 User Interaction

The user controls the solution process in interactive optimization. The user addresses the hard-to-quantify issues, specifies or fixes problem parameters, applies the algorithms to the problem or pieces of the problem, and decides when the solution is acceptable. High-level strategic decisions are made by the user.

1.1.3 Interactive Interface

The interactive interface is the medium of communication between user and computer. The interface must present parameters and solutions

in a form that exploits the user's ability to work toward an acceptable solution. The best interface for this purpose is graphics[2]. Graphics also efficiently captures large amounts of information.

1.1.4 Interactive Optimization Methodology

An interactive optimization problem is typically intially solved using developed algorithms. Next, the user utilizes interactive graphics to analyze and tune the solution by modifying or fixing certain input parameters or parts of the solution. Finally, the algorithms are reapplied to the entire problem or pieces of the problem. The process is repeated until the user is satisfied with the solution.

2.0 DESIGN OF THE MATHEMATICAL MODEL

This section constructs the underlying mathematical model. The model is computationally difficult but has some structure which suggests the illustrated relaxation (see Fisher[3],[4] and Geoffrion [5]). An algorithm to solve the relaxed model is presented. Computational results are discussed in Section 3.0.

2.1 Problem Formulation

Define:

S = set of naval ships

R = set of strategic requirements

a_s = set of activities ship s can process

r_a = set of activities of requirement r

$$x_{sr} = \begin{cases} 1 \text{ if requirement is r assigned to ship s} \\ 0 \text{ otherwise} \end{cases}$$

$$y_{sqr} = \begin{cases} 1 \text{ if ship s processes requirement q immediately before processing requirement r.} \\ 0 \text{ otherwise} \end{cases}$$

The primary objective is to satisfy all the requirements. To accomplish this objective the model can maximize the number of requirements satisfied. If the model is solved to optimality and not all requirements are satisfied, then it is not possible to satisfy all requirements. Thus, the objective is

$$\text{MAX} \quad \sum_{S} \sum_{R} x_{sr}$$

Ship s can process only one requirement at a time. Thus, if requirement r is performed by ship s there must be exactly one requirement which preceeds requirement r on ship s. If requirement r is not performed by ship s then there is no requirement which precedes requirement r on ship s. This constraint can be expressed as

$$x_{sr} - \sum_{r=1}^{R} y_{sqr} = 0 \quad q=1,\ldots,R \quad (1)$$

If ship s does not have the characteristics necessary to process requirement r or is unavailable because of overhaul commitments, then x_{sr} is set to zero and does not appear in the problem. If requirement r cannot immediately follow requirement q because it would not allow ship s enough time in port or because r is prohibited by the planner from following q, then y_{sqr} is set to zero and does not appear in the problem. Note that the definition of y_{sqr} prohibits ship s from having two requirements assigned to it at the same time.

If there is only one ship in the scenario, the problem is equivalent to a longest path problem in a directed acyclic network. The network is illustrated in the following example.

2.1.1 Single Ship Network Example

Consider an example problem having five requirements, denoted as R1 through R5. The required processing time is represented by the length and horizontal position of the bars in Figure 1. The characteristics needed for each requirement are listed inside the corresponding bar.

Suppose a ship s has the capability to perform activities A1,A3,A4, and A5. The longest path network for ship s is shown in Figure 2. There is an arc from the completion of one requirement (e.g.,R1) to the beginning of another requirements (e.g.,R5) if it is acceptable for the ship to perform the two requirements in sequence. For example, the arc from the end of R5 to the beginning of R6 indicates that ship s can satisfy R5 and then have the required time in port and transit time to satisfy R6.

If it is not acceptable for a ship to satisfy two requirements (e.g.,R2 and R5) then no arc is constructed between the two requirements. Note requirement R6 is not eligible for the network, since ship s cannot perform activity A2 of this requirement.

Finding the maximum number of requirements which can be satisfied by ship s is equivalent to finding the longest path in the directed acyclic in Figure 2.

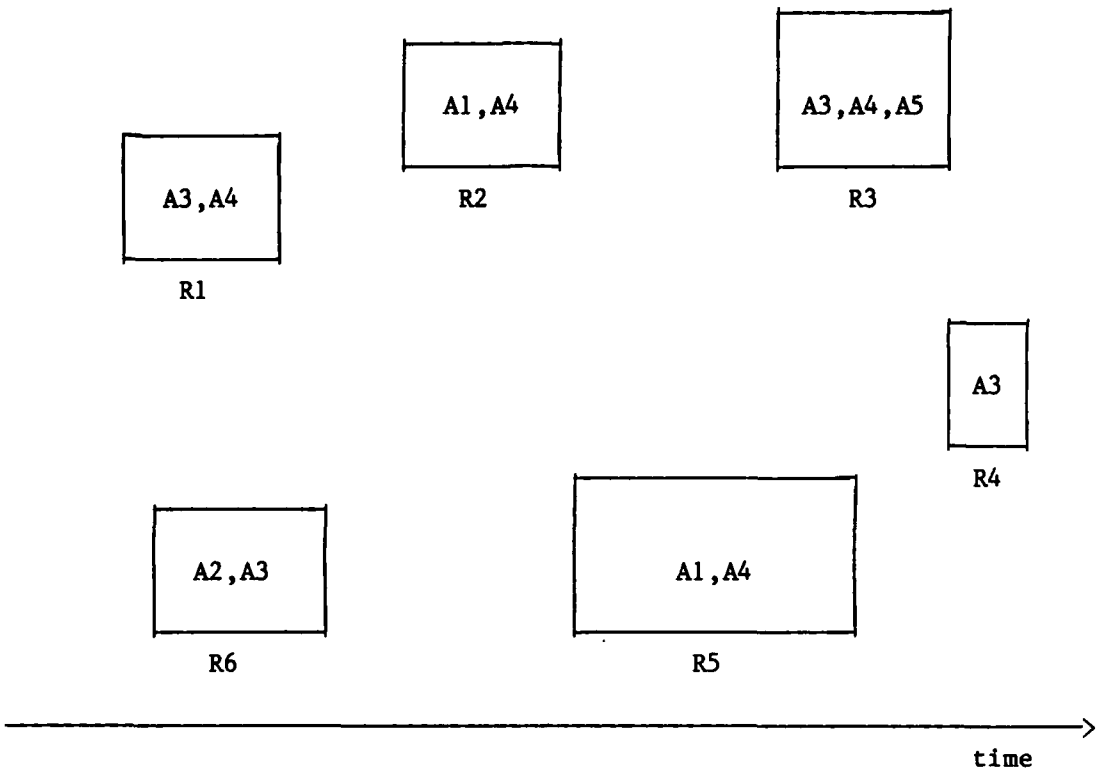


FIGURE 1: Bar Chart Representing
Times for Requirements

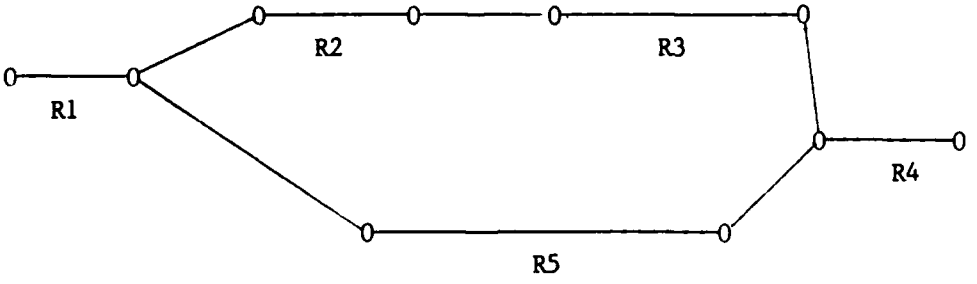


FIGURE 2: Longest Path Network
For Example of Figure 1

2.2 Formulation For Multiple Ships

The single ship formulation can be extended to include the entire fleet. Define P_s as the constraint set (constraint (1) from section 2.0) for the longest path network for ship s. The objective function for the entire fleet is

$$\text{MAX} \quad \sum_S \sum_R x_{sr}$$

SUBJECT TO

$$P_1 \qquad P_2 \qquad \ldots\ldots\ldots \qquad P_s$$

Additionally, each requirement can be assigned to only one ship, giving

$$\sum_S x_{sr} = 1 \quad r=1,\ldots,R \quad (2)$$

Thus, the multiple ship problem is a set of longest path problems linked by the constraint (2). While each individual ship problem can be solved very efficiently, constraint (2) cause the overall problem to be a very large integer program. The complexity of the formulation, compounded by the size of the actual Atlantic Fleet problem, necessitates some form of relaxation. The natural approach is to relax constraint (2).

2.3 Model Relaxation

If the constraint prohibiting assignment of a requirement to more than one ship is relaxed, the problem decouples into S longest path

problems, one for each ship. The constraints (2) can be brought into the objective function, yielding the following formulation:

$$\text{MAX} \quad \sum_S \sum_R x_{sr} + \sum_R \lambda_r \sum_S x_{sr}$$

SUBJECT TO

P_1 P_2 ..... P_s

where λ_r is a constant whose value must be specified. The problems then decouple into a longest path problem for each ship where the cost on the dashed arcs in Figure 2 are $(1 + \lambda_r)$. Each of these problems is easily solved. However, because the networks are solved independently, some requirements may be assigned to more than one ship, and some requirements may not be assigned to any ship. The following section presents an algorithm to try to overcome these difficulties.

2.4 Relaxation Algorithm

Step 1. Set $\lambda_r = 1$ for r=1,...,R (there is initially no preference between requirements). Solve the longest path problem for each ship.

Step 2. If the resulting schedules are individually feasible, and every requirement is covered, a complete schedule is generated. STOP.

Step 3. If some requirement r is not satisfied, it is posible to determine how much to increase λ_r in order to force requirement r into an

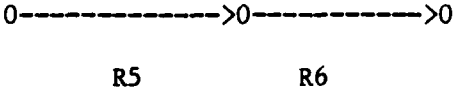
eligible ship's longest path solution. The following steps are required:

1. For a ship s which has all characteristics necessary to satisfy requirement r, solve the longest path problem. Let the objective function solution to this problem = Z.
2. For the same ship, solve the longest path problem up to the start of requirement r. Let the objective function solution to this problem = Z1.
3. For the same ship, solve the longest path problem from the completion time of requirement r to the end of the time horizon. Let the objective function solution to this problem = Z2.
4. Increase λ_r to Z - (Z1+Z2) + e, where e is > 0. With this increase in λ_r, it is more attractive to include requirement r in the ship s longest path solution.
5. Repeat steps (1) through (4) for each uncovered requirement and for each compatible ship. Select the minimum λ_r for each requirement. This will attempt to force requirement r into at least one ship's schedule, with minimal disruption of the existing schedule.

Example:

Define the longest path network for ship 1 as the network in Figure 2.

Define the longest path network for ship 2 as



Define the initial set of values as

$$\lambda_1=\lambda_2=\lambda_3=\lambda_4=\lambda_5=\lambda_6 = 1.$$

The longest path solution for ship 1 is Z = 4 (with requirements R1,R2,R5,R4). The longest path solution for ship 2 is Z = 2 (with requirements R5, R6). R3 is not included in either ship's initial solution. We wish to calculate the λ_3 necessary to force R3 into a longest path solution.

Solving the ship 1 longest path subnetwork up to the start time of R3 gives

Z1 = 2 (with solution R1,R2).

Solving the ship 1 longest path subnetwork from the completion time or R3 gives

Z2 = 1 (with solution R4).

Then

$$\lambda_3 = Z - (Z1 + Z2) + e = 1 + e.$$

Ship 2 is not eligible to process R3, so no subnetwork calcuations are necessary.

With $\lambda_3 = 1 + e$ $(e > 0)$ it is more attractive to include R3 in the longest path solution of ship 1, giving the ship 1 solution of R1,R2,R3,R4.

Step 4. Using the new λ_r 's as edge weights, solve the longest path network for each ship. GOTO Step 2.

This algorithm may assign requirements to multiple ships. These requirements can be removed easily:

1. Rank the ships claiming multiple requirements by utilization (precent of time processing requirements).
2. Remove a multiple requirement from the highest utilized ship.
3. Update the utilization of the ship removed of the requirement.
4. If no other requirements are assigned to multiple ships, STOP. Otherwise GOTO (1).

It is possible that no solution exists which will cover all requirements for a given fleet scheduling problem instance. In this case the planner must decide what parameters to change, including modifying overseas requirements, changing overhaul schedules, or overriding ship-in-port time constraints.

3.0 Algorithm Performance Results

Since the actual Atlantic Fleet scheduling is classified, randomly generated problems with characteristics similar to those of the actual fleet scheduling were used to measure the performance of the algorithm. In most cases the initial longest path solutions left a few requirements (typically 10%) unscheduled. However, performing 5-20 iterations of the algorithm ususlly scheduled all requirements that could be scheduled. Requirements still unscheduled after iterating were usually infeasible.

3.1 Level of Human Control

The user requires the ability to control the following tasks in the interactive optimization:

1. Apply the solution algorithm at any time.
2. Fix a requirement to a ship.
3. Remove a requirement from a ship.
4. Determine the set of requirements that are eligible for assignment to a ship.
5. Determine the set of ships that are eligible to process a requirement.
6. Change the restrictions regarding time in port in order to generate an acceptable solution.

These options must be incorporated in a frame work which allows the user to easily work towards an acceptable solution. The following section illustrates the interactive optimization model using an example problem.

Table 1

Example Problem Ship Data

First ship	available	Overhaul months start	finish	activities
1	1	after horizon		2,4,5,6
2	2	12	20	2,3,6,7,8
3	2	after horizon		1,2,3,6,9
4	1	after horizon		3,7,8,9
5	1	21	30	1,4,7,9
6	8	24	36	2,4,5,6,7
7	6	after horizon		1,2,3,4,5,6,7,8,9
8	9	after horizon		3,5,7
9	1	6	16	1,2,3,9
10	3	after horizon		1,3,5,6,7,8,9
11	8	after horizon		1,2,4,6,8
12	1	4	20	1,2,4,5,7,8
13	1	12	18	2,3,8,9
14	7	28	35	1,2,3,4,5,6,7,8,9
15	1	25	33	6,7,8

Table 2

Example Problem Requirements Data

Requirement	Start	Finish	Activities
1	2	8	3,9
2	6	9	1,2,5,8
3	10	12	4,6
4	9	14	1,2,4
5	7	9	2,5
6	1	10	1
7	12	18	5,6,7
8	11	17	1,3
9	19	20	6,8
10	14	24	4,7
11	11	18	1,9
12	5	11	2
13	21	27	3,6
14	21	28	6
15	13	23	7
16	14	22	4
17	20	27	2,8
18	16	23	6
19	3	11	7
20	15	24	4,9
21	30	36	5,8
22	15	21	2,4,6
23	22	29	1,3,6
24	18	24	9

4.0 Interactive Interface and Solution Process (Example)

The interactive optimization model for this problem was developed on a Chromatics 7900 high resolution color graphics computer. The model utilizes an editor screen with a light pen to control user options. The system also employs a screen which graphs the performance of the algorithm, and a screen which illustrates the current fleet schedule.

The interactive optimization methodology is illustrated using a 15 ship, 24 requirement example problem. Table 1 presents the ship date; Table 2 presents the requirement characteristics. A 36 month planning horizon is used for this problem, with months designated as 1 through 36.

The example problem also has the following properties:

1. For strategic reasons requirement 17 has been fixed to ship 11, and requirement 16 has been fixed to ship 12.
2. Ship 7 has been overseas for the previous two Christmas seasons, and must be home for the next Christmas season (defined as months 11 and 12).
3. Ship 15 must be home for at least 24 months of the planning horizon.

The user begins the solution process by assigning requirement 6 to ship 11 and requirement 16 to ship 12, as required by the problem. Next, several iterations of the algorithm are run to provide a good initial fleet schedule. The performance of the algorithm is given in Figure 3, illustrating that five iterations were required to assign all requirements to ships. Figure 4 shows the fleet schedule for the fifth iteration.

The user now assesses the quality of this solution, and uses the ship editor for "what if" analyses. Most of the ship schedules are fairly balanced. However, ship 15 is not home for at least 24 months of the planning horizon, as specified by the problem. Another candidate for analysis is ship 14, which makes two overseas trips in less than one year.

The user can manipulate ship schedules to improve the solution. For example, either requirement 18 or 19 should be assigned from ship 15 to another ship. Further processing may be required to balance ship workload or minimize the number of overseas trips because of this reassignment.

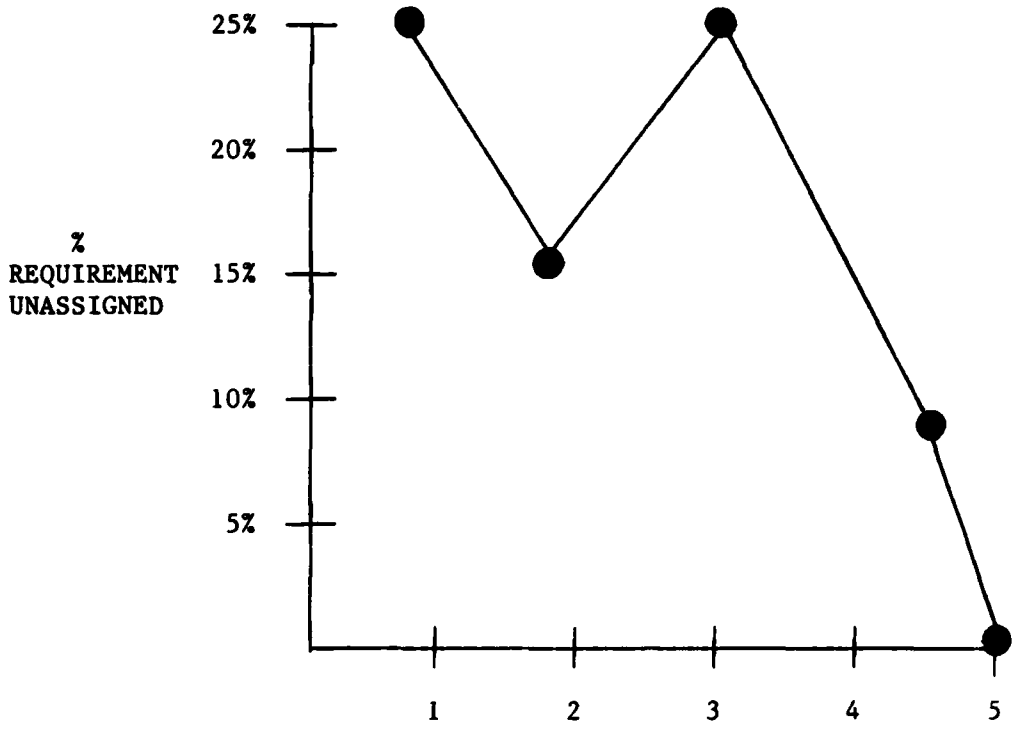


Figure 3

ALGORITHM PERFORMANCE

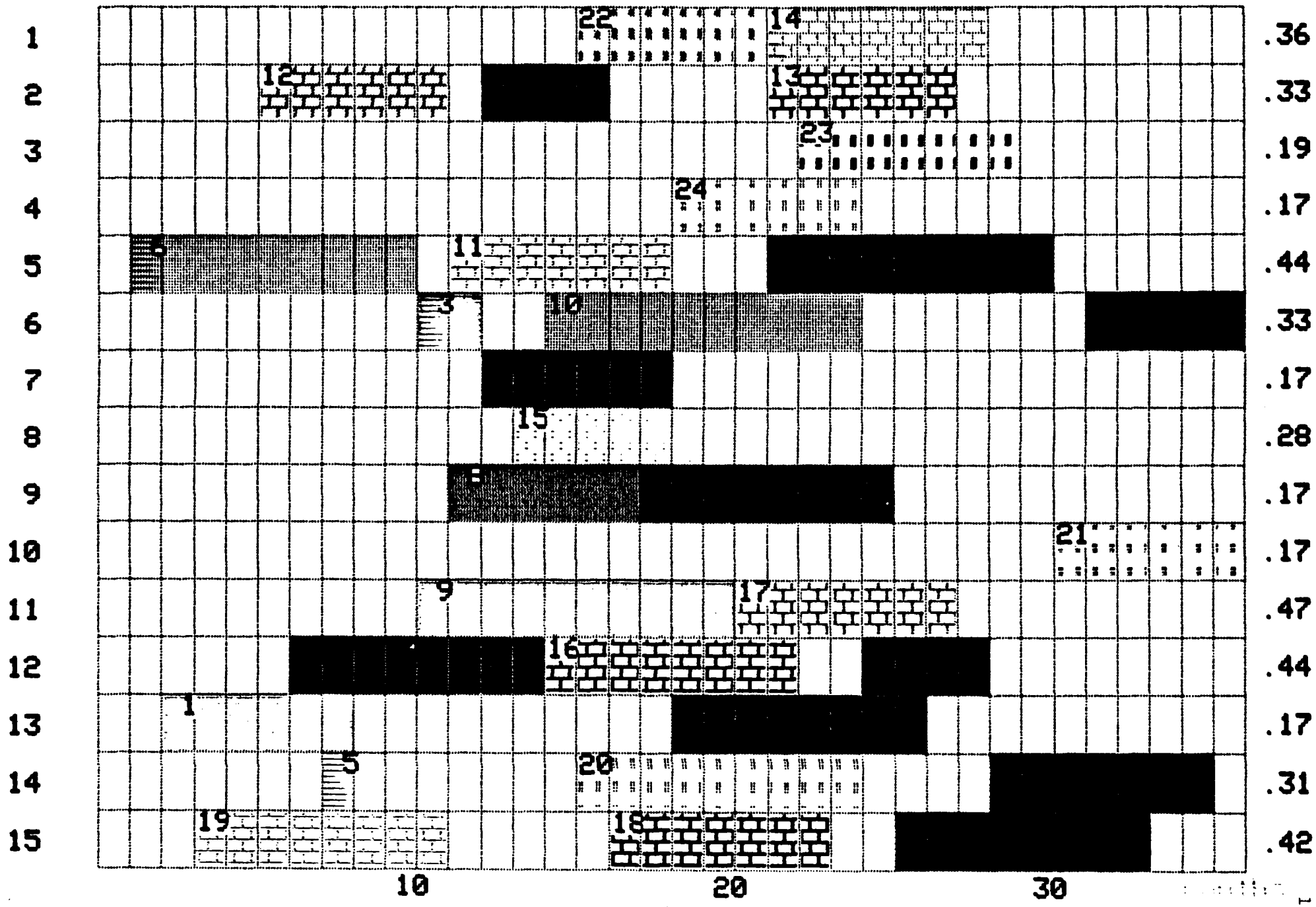

ATLANTIC FLEET SCHEDULE
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
.36
.33
.19
.17
.44
.33
.17
.28
.17
.17
.47
.44
.17
.31
.42
10
20
30
22
14
12
13
23
24
11
3
10
15
21
9
17
16
1
5
20
19
18

Figure 4

5.0 Summary

The Atlantic Fleet ship scheduling problem is effectively approached using an iteractive optimization approach. The problem is numerically complex, requiring the computer to solve numerous network problems. The problem also contains issues which cannot be effectively quantified, and thus should be addressed by an experienced naval planner.

The interactive optimization system developed for the problem combines an iterative network model with a flexible man-machine graphics interface. This system allows the user to generate a good initial fleet schedule using the network algorithms, and to interactively improve the solution by addressing the difficult-to-quantify issues.

[1] G. B. Dantzig and D. B. Fulkerson, "Minimizing the Number of Tankers to Meet a Fixed Schedule," Naval Research Logistics Quarterly, (1954).

[2] Tufte, E. R., The Visual Display of Quantitative Information, Graphics Press, Chesire, Ct., 1983.

[3] M. L. Fisher, "The Langrangean Relaxation Method for Solving Integer Programming Problems," "Management Science" 27 (1981) 1-18.

[4] M. L. Fisher, "An Applications Oriented Guide to Lagrangean Relaxation," Interfaces 15(2) (1985) 10-21.

[5] A. M. Geoffrion, "Lagrangean Relaxation and its Use in Integer Programming," Mathematical Programming Study No. 2 (1974) 82-114.

END

1-87

DTIC